ECE 6882 – Reinforcement Learning

Spring 2023 – The George Washington University

Osama Yousuf – osamayousuf@gwu.edu

Project 1 – Multi-Armed Bandit

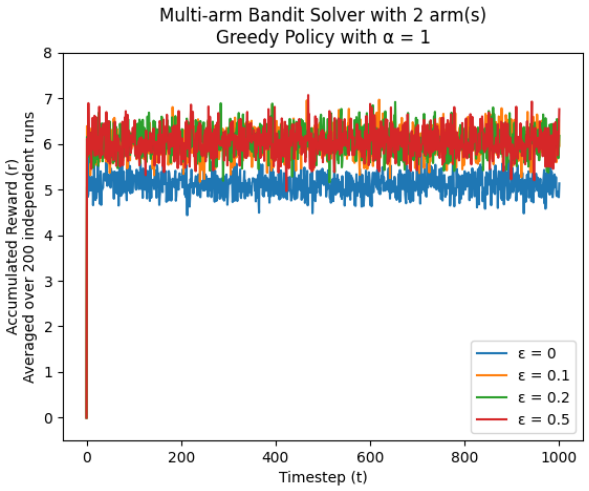
In this project, I have implemented two different policies to solve the multi-armed bandit problem, and I have carried out experiments over various algorithmic hyperparameters to establish a sense of which algorithm performs better than others. I used environment 3 (written by myself). Supplemental code files are provided as an accompanying archive.

Experiments were conducted over a total of a 1000 timesteps, and repeated for a total of 200 independent runs for statistical significance since the reward received per timestep is a random variable.

Experiment a

In this part, I implemented the ε-greedy policy. The initial Q values were fixed at $Q(a^1) = Q(a^2) = 0$, and the learning rate α as well as the ε parameter was varied. The following sections present results on four such learning rates.

1) $\alpha = 1$

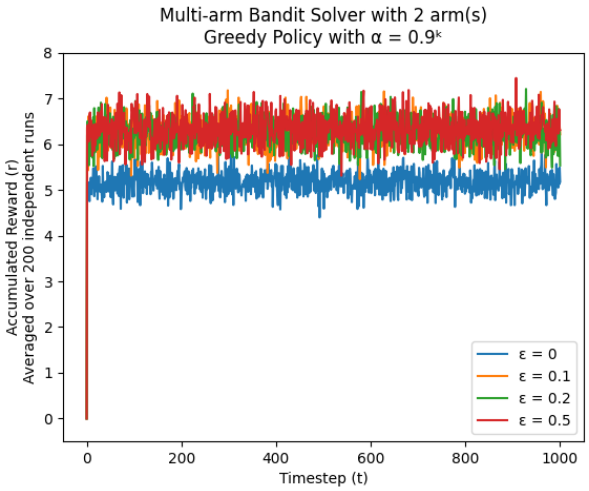


ε-greedy	Average of action value $Q(a^1)$ of 100 runs	True action value $Q^*(a^1)$	Average of action value $Q(a^2)$ of 100 runs	True action value $Q^*(a^2)$
$\varepsilon = 0$ (greedy)	4.23803222	5	-7.72449087	7
$\varepsilon = 0.1$	2.41732452	5	3.89244477	7
$\varepsilon = 0.2$	3.19641559	5	4.33671683	7
$\varepsilon = 0.5$ (random)	4.41417365	5	5.71348755	7

A learning rate of $\alpha = 1$ corresponds to a relatively higher amount of weightage being given to the correction term that is seen. In this case, when $\varepsilon = 0$, we see that $Q(a^2)$ takes on a significantly negative value. This makes sense because action 2 was only chosen 5% of the time - 11,526 times out of 200,000 (200 independent runs, each having 1000 timesteps). We conclude that $\varepsilon = 0$ policy was unable to successfully maximize the expected reward function, as is also evident from the corresponding graph.

In all other cases, $Q(a^2) > Q(a^1)$, which indicates that the agent will choose action 2 more frequently than action 1. Based on this, it can be concluded that the ε-greedy policy is able to solve the two-armed multi-bandit problem since it ultimately ends up choosing the correct action 2 (which maximizes the reward). Finally, we see that $\varepsilon = 0.5$ performs the best when it comes to approximating the true Q^* values for both the actions, but in terms of achieving the best accumulated reward, $\varepsilon = 0.2$ performs the best, achieving an average value of 6.05.

2) $\boldsymbol{\alpha = 0.9^k}$



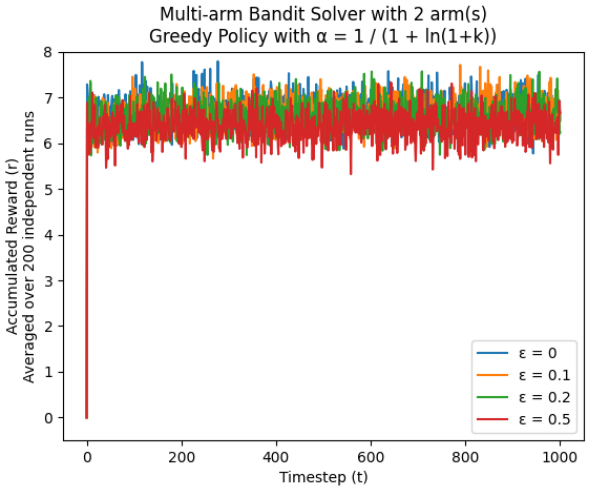
ε-greedy	**Average of action value $Q(a^1)$ of 100 runs**	**True action value $Q^*(a^1)$**	**Average of action value $Q(a^2)$ of 100 runs**	**True action value $Q^*(a^2)$**
$\varepsilon = 0$ (greedy)	4.38396068	5	-2.56472551	7
$\varepsilon = 0.1$	3.44925138	5	5.20521596	7
$\varepsilon = 0.2$	4.19972723	5	5.65442969	7
$\varepsilon = 0.5$ (random)	4.81680875	5	6.51517737	7

A learning rate of $\alpha = 0.9^k$ corresponds to an exponential decrease with respect to the timestep. Higher the timestep, smaller is the value of α, and so we assign smaller weights to the correction term when updating Q values as time increases. Again, we see that when $\varepsilon = 0$, $Q(a^2)$ takes on a negative value. We conclude that $\varepsilon = 0$ policy was unable to successfully maximize the expected reward function, as is also evident from the corresponding graph.

In all other cases, $Q(a^2) > Q(a^1)$, which indicates that the agent will choose action 2 more frequently than action 1. Based on this, it can be concluded that the ε-greedy policy is able to solve the two-armed multi-bandit problem since it ultimately ends up choosing the correct action 2 (which maximizes the reward).

Compared to $\alpha = 1$, all ε-values perform better in approximating true Q^* values. In addition, we see that $\varepsilon = 0.5$ again performs the best when it comes to approximating the true Q^* values for both the actions. In terms of achieving the best accumulated reward, $\varepsilon = 0.5$ also performs the best, achieving an average value of 6.32.

3) $\boldsymbol{\alpha = \frac{1}{1+ln(1+k)}}$

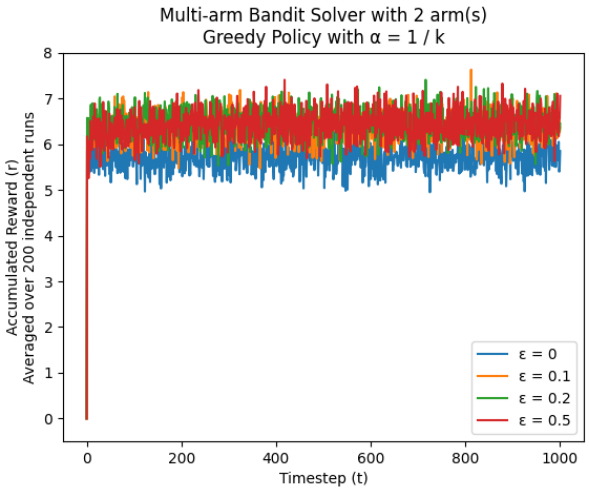


ε-greedy	**Average of action value $Q(a^1)$ of 100 runs**	**True action value $Q^*(a^1)$**	**Average of action value $Q(a^2)$ of 100 runs**	**True action value $Q^*(a^2)$**
$\varepsilon = 0$ (greedy)	-0.26929882	5	5.97408814	7
$\varepsilon = 0.1$	4.44772347	5	6.64791538	7
$\varepsilon = 0.2$	4.64449209	5	6.80070588	7
$\varepsilon = 0.5$ (random)	4.81086777	5	6.90994588	7

A learning rate of $\alpha = \frac{1}{1+\ln(1+k)}$ corresponds to a logarithmic decrease with respect to the timestep. Higher the timestep, smaller is the value of α, and so we assign smaller weights to the correction term when updating Q values as time increases, though the decrease isn't as harsh as the exponential decaying learning rate. Contrary to previous cases, here we see that $Q(a^2) > Q(a^1)$ for all cases, which indicates that the agent will choose action 2 more frequently than action 1. Based on this, it can be concluded that the ε-greedy policy is able to solve the two-armed multi-bandit problem since it ultimately ends up choosing the correct action 2 (which maximizes the reward). Additionally, we conclude that this logarithmically decreasing learning rate is more robust and might actually be a better choice than the other learning rates, since it is able to determine the action with the higher average reward in more cases of varying ε values.

Compared to previous learning rates, all ε-values perform better in approximating true Q^* values. In addition, we see that $\varepsilon = 0.5$ again performs the best when it comes to approximating the true Q^* values for both the actions. However, $\varepsilon = 0$ performs the best in terms of achieving the best accumulated reward, achieving an average value of 6.7.

4) $\boldsymbol{\alpha = \frac{1}{k}}$



ε-greedy	**Average of action value $Q(a^1)$ of 100 runs**	**True action value $Q^*(a^1)$**	**Average of action value $Q(a^2)$ of 100 runs**	**True action value $Q^*(a^2)$**
$\varepsilon = 0$ (greedy)	2.46762614	5	-0.01681477	7
$\varepsilon = 0.1$	3.93538698	5	5.91793999	7
$\varepsilon = 0.2$	4.65680694	5	6.36024853	7
$\varepsilon = 0.5$ (random)	4.84397095	5	6.81058607	7

A learning rate of $\alpha = \frac{1}{k}$ corresponds to a simple polynomial decrease with respect to the timestep. Higher the timestep, smaller is the value of α, and so we assign smaller weights to the correction term when updating Q values as time increases. Similar to the constant and exponentially decreasing case, we see that when $\varepsilon = 0$, $Q(a^2)$ takes on a negative value. We conclude that $\varepsilon = 0$ policy was unable to successfully maximize the expected reward function, as is also evident from the corresponding graph.

In all other cases, $Q(a^2) > Q(a^1)$, which indicates that the agent will choose action 2 more frequently than action 1. Based on this, it can be concluded that the ε-greedy policy is able to solve the two-armed multi-bandit problem since it ultimately ends up choosing the correct action 2 (which maximizes the reward). Compared to $\alpha = 1$, all ε-values perform better in approximating true Q^* values. In addition, we see that $\varepsilon = 0.5$ again performs the best when it comes to approximating the true Q^* values for both the actions. In terms of achieving the best accumulated reward, $\varepsilon = 0.2$ performs the best in terms of achieving the best accumulated reward, achieving an average value of 6.45.

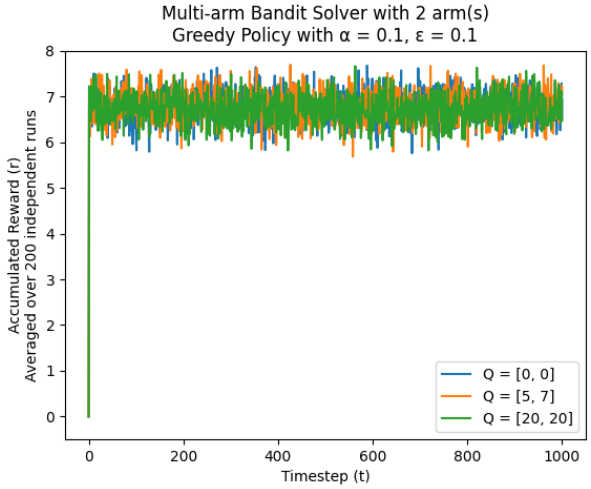
Based on my simulation results, I conclude that the logarithmically decreasing learning rate, $\alpha = \frac{1}{1+ln(1+k)}$, exhibited the best performance overall. Moreover, a combination of this learning rate, $\alpha = \frac{1}{1+ln(1+k)}$, along with an ε-value of 0, led to the highest average accumulated award value of 6.7. The following table lists the top 5 (α, ε) combinations in the order of decreasing average accumulated reward (highest first).

Learning Rate α	Epsilon ε	Average Accumulated Reward
$\frac{1}{1+ln(1+k)}$	0 (greedy)	6.70277392724691
$\frac{1}{1+ln(1+k)}$	0.1 (in-between)	6.636603869133994
$\frac{1}{1+ln(1+k)}$	0.2 (in-between)	6.574094706601989
$\frac{1}{k}$	0.2 (in-between)	6.453961246419116
$\frac{1}{k}$	0.5 (random)	6.412377927626179

If we were to rank the learning rates as best to worst ranked on the average accumulated reward, the logarithmic case is the best, followed by the inverse polynomial, then the exponential, and then finally the constant being the worst.

Experiment b

In this part, I used the same ε-greedy policy. The learning rate α as well as the ε parameter was fixed at 0.1, whereas the initial Q values were varied. Three combinations were explored – one with Q values fixed at 0, another with Q values set to exactly the true Q^* values for each of the two actions, and the last with Q values significantly greater than the true Q^* values.



Initial Q values	Average of action value $Q(a^1)$ of 100 runs	True action value $Q^*(a^1)$	Average of action value $Q(a^2)$ of 100 runs	True action value $Q^*(a^2)$	Average Accumulated Reward
$Q = [0, 0]$	4.5674671	5	6.79130827	7	6.769422810181119
$Q = [5, 7]$	4.61838073	5	6.87876857	7	6.773625552948703
$Q = [20, 20]$	4.50707796	5	6.77921375	7	6.713822060980149

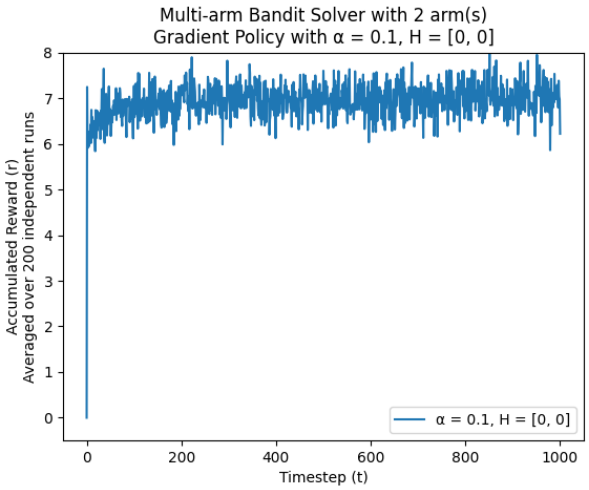
From the graph and the corresponding table, it can be seen that the agent picks action 2 more frequently than action 1 in all cases, so the two-armed bandit problem is technically solved in all cases regardless of the choice of the initial Q values since the agent has discovered that action 2 has a higher expected value. However, there are still differences which can be observed if we look at the frequency of each action being chosen and the corresponding average accumulated reward against each initial Q value. We see that the second case, where Q values are initially set to the exact Q^* true values i.e. $[5, 7]$ attains the best performance i.e., the highest average accumulated reward overall compared to the other cases. For the other two cases, the average accumulated reward is lower, with the extreme case of $[20, 20]$ having the lowest performance (though still better than hyperparameters explored in **experiment a**).

This can be interpreted in the following way: When we have initial values of $[0, 0]$, the agent does not perform exploration during initial timesteps. The agent inefficiently spends time in pulling whichever arm is selected first, even if it is sub-optimal. On the other end, when we have initial values of $[20, 20]$, the agent performs too much exploration. This is even more inefficient and results in the agent taking much longer to converge to the optimal arm (which is action 2).

Under the given (α, ε) combination of $(0.1, 0.1)$, having initial values of $[5, 7]$ leads to the best performance, which can be interpreted as leading to the perfect balance of exploration and exploitation. It must be noted that in a realistic scenario, true Q^* values can not be known, and so these results point to the importance of performing extensive hyperparameter optimization to fully understand dynamics of the problem under investigation.

Experiment c

In this part, I implemented a gradient-based softmax policy for solving the multi-armed bandit problem. The learning rate α was fixed at 0.1, and the initial preference for each action was set to $H(a^1) = H(a^2) = 0$. The following figure gives the accumulated reward of this policy as time proceeds averaged over 200 independent runs for capturing statistically significant trends.



This gradient-based policy performs better than all cases of the epsilon-greedy policy explored in both, experiment a as well as experiment b, achieving the highest accumulated average reward value of 6.9 – extremely close to the true $Q^*(a^2)$ value of 7. The final preference values (averaged) were $H = [H(a^1), H(a^2)] = [-3.11332582,\ 3.11332595]$, leading to softmax selection probabilities of $[0.00255548, 0.99744452]$ respectively.

The following table summarizes statistics about this gradient-based policy and the greedy-policy with $\alpha = \varepsilon = 0.1, Q = [0, 0]$ (from Experiment 2).

Policy	**Policy Parameters**	**Average Accumulated Reward (Highest to lowest)**	**% of choosing $a^* = a^2$ (higher, the better)**	**Experiment**
Gradient	α = 0.1 H = [0, 0]	6.922368016509635	**97.05 %**	C
ε-greedy	α = 0.1 ε = 0.1 Q = [5, 7]	6.773625552948703	88.56 %	B
	α = 0.1 ε = 0.1 Q = [0, 0]	6.769422810181119	87.91 %	B
	α = 0.1 ε = 0.1 Q = [20, 20]	6.713822060980149	87.12 %	B
	α = 1 / (1 + ln(1+k)) ε = 0 Q = [0, 0]	6.70277392724691	85.91 %	A

It can thus be concluded that on the given problem, the gradient policy from **experiment c** outperforms the ε-greedy policy from **experiments a and b** significantly. This makes sense since the gradient-based policy takes a much more nuanced approach when exploring the actions as it sets the probabilities of taking each action via the gradient of the expected reward of each arm. This allows the agent to explore less promising actions with a lower probability while still being able to find the optimal action with high probability, in turn balancing exploration and exploitation phases efficiently. In contrast**,** ε-greedy is a simpler approach to the problem, and was less effective at exploring the action space under the investigated hyperparameters, as evident from the table above. The % of choosing the optimal action a^2 has a direct correlation with how high the average accumulated reward is. A higher % means that the optimal action i.e., the optimal lever was pulled more than the suboptimal lever on average over all the experiments.